# Self-Adaptive Discovery Mechanisms for Optimal Performance in Fault-Tolerant Networks


## Abstract

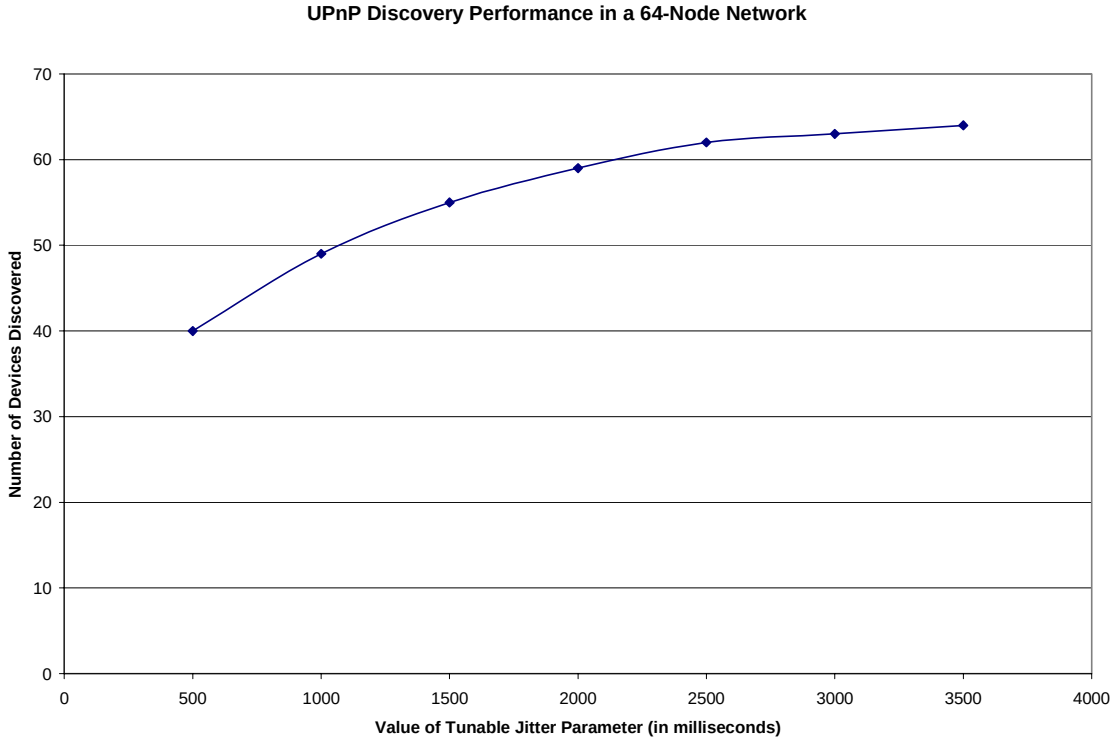
Emerging designs for fault-tolerant systems rely on discovery-based component architectures to enable self-organizing and self-healing systems. The underlying service discovery and composition technologies for such systems include mechanisms that permit the network to continue to function even as its configuration changes continuously over time. Unfortunately, in very dynamic environments, such as found in military applications, little is known about the behavior and performance of emerging service discovery protocols. Many aspects of the performance of these protocols appear highly sensitive to parameter settings whose optimum values depend upon the composition of the network. While such parameters may be manually tuned in relatively small, static environments, their management in large-scale, highly dynamic environments requires real-time measurement and control. *NIST proposes to research, design, evaluate, and implement self-adaptive algorithms to improve the performance of service discovery and composition technologies for use in fault-tolerant networks.* Such algorithms will continuously and dynamically measure the composition of a fault-tolerant network and adjust appropriate parameter settings for relevant protocols. During the research phase, NIST will examine various service discovery and composition technologies emerging in industry and academe to select several for further development in subsequent phases of the project. During the design and evaluation phase, NIST will design self-adaptive algorithms to control the execution of selected protocols, and will conduct simulation experiments to evaluate performance improvements and costs associated with the proposed algorithms. In the implementation stage, NIST will augment publicly available reference code for service discovery protocols to include the best algorithms identified during the evaluation phase of the project. NIST will also propose the inclusion of these self-adaptive algorithms within emerging public specifications in the Internet Engineering Task Force (IETF) and other appropriate forums. *Results from this proposed work could lead to significant improvements in the design of service discovery and composition technologies to more readily accommodate the self-adapting algorithms necessary for use in large-scale fault-tolerant networks*.


## Rationale

Future military C4I systems will encompass network-centric elements that continuously appear and disappear as they move and reorganize in the face of physical and cyber attacks. To cope with such dynamics, emerging C4I architectures, responding to Joint Vision 2010, are adopting flexible component architectures [OWING00] that rely on the performance of service discovery protocols to enable self-organizing and self-healing systems. These discovery protocols, now being defined in various industry forums [JINI00, SLP99, SSDP99, UPNP00], aim to provide an underlying foundation for robustness in future distributed systems. In general, discovery protocols enable network nodes to come and go at will, to discover and rendezvous with other nodes, to determine

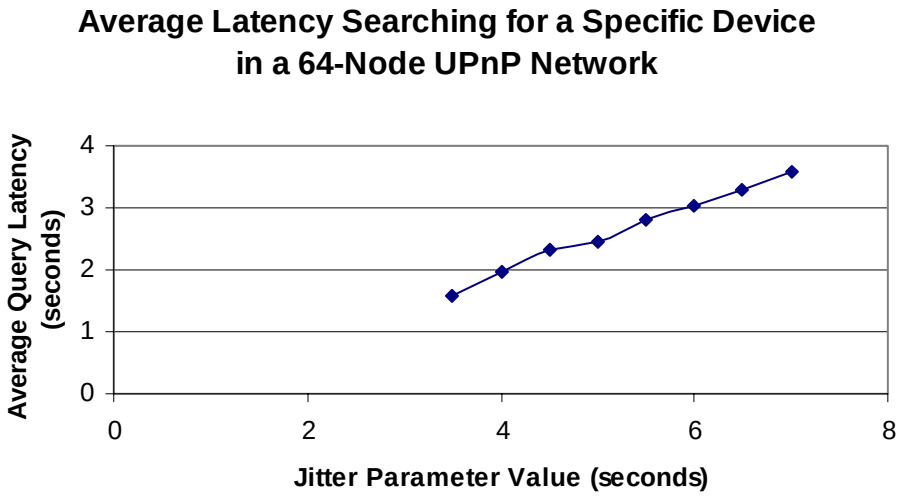
available networking services, and to connect to and use the available services that meet requirements. Because such capabilities will provide a basis for constructing future fault-tolerant network services, researchers in the NIST Information Technology Laboratory have begun to study the properties of the most significant dynamic discovery protocols currently under design.

Our initial research, and related measurements, uncovered some unfortunate properties that appear to be inherent in the designs of the emerging dynamic discovery protocols. In outline, the performance of such protocols depends upon parameter settings whose optimum values also depend upon the composition of the network, where the composition of the network can of course change continuously. Such circular dependency can lead routinely to situations where the protocols yield extremely poor performance. As an example, consider the universal plug and play (UPnP) protocol [UPNP00], whose performance we measured through the use of publicly available code from Intel. <http://developer.intel.com/ial/upnp/>




The graph above shows that the ability of UPnP to discover devices on a network depends upon using an acceptable minimum value for a tunable jitter parameter within UPnP. The jitter parameter defines an upper bound for responses from network nodes, where individual network nodes randomly select a response time from a range between immediate response and the value of the jitter parameter. Our measurements demonstrate that setting the jitter parameter too low prevents the protocol from discovering all the nodes in the test network, probably because the discovering node cannot handle the resulting traffic intensity. Once the value of the jitter parameter reaches or exceeds an acceptable minimum setting, then all the devices can be discovered. Of course, determining the acceptable minimum jitter value depends upon having some idea about

the number of nodes on the network. But of course it is the discovery protocol that provides an idea about the number of nodes on the network. How might this conundrum be resolved? The results of this experiment appear to suggest that choosing a sufficiently large jitter value might enable all devices to be discovered for networks up to some maximum size. Unfortunately, setting the jitter value too high leads to some other performance problems, as illustrated in the next graph.




In the graph above, we show how the value of the jitter parameter affects the average response time when searching for a specific node (by unique identifier) on our UPnP test network. Since a device is required to randomize its response to a query, where the upper bound is based on the jitter value, the average response time will be around one-half the jitter value. This means that the higher the jitter value, the higher the average response time. Similar effects can be seen when searching for devices by type, although the response latency in those cases also depends upon the number of devices of that type found on the network. While our examples here used UPnP as an illustration, similar situations appear in other dynamic discovery protocols being considered as a basis for self-organizing and self-healing networks.

These initial findings support our claim that, to achieve optimal performance in highly dynamic fault-tolerant networks, emerging discovery protocols will likely require self-adapting algorithms that continuously adjust tunable parameter values based upon dynamic measurement of network conditions. Our example also suggests how difficult it might be to design and evaluate appropriate algorithms because significant interactions can occur among different protocol features based on adjusting a single tunable parameter within a protocol. In fact, r*esults from this proposed work could lead to significant improvements in the design of service discovery and composition technologies to more readily accommodate the self-adapting algorithms necessary for use in large-scale fault-tolerant networks.*

## Technical Approach

In thumbnail, our technical approach will be staged into four phases: (1) research, (2) design and evaluation, (3) implementation, and (4) influence. During the research phase we will examine selected dynamic discovery technologies to identify candidate protocol mechanisms where performance depends jointly on tunable parameter settings and network conditions. To the extent practical, we will support our analyses with measurements made against existing prototype implementations of the protocols. We will analyze and catalog the mechanisms and related dependencies. At the same time, we will search the literature for existing self-adaptive algorithms proposed and evaluated by other researchers. For example, the adaptive retransmission time-out control algorithms proposed by Van Jacobson, Karn, and others would be in this class. We expect to find other such adaptive algorithms associated with the design of multicast protocols and redundant distributed-caching and directory systems. We will attempt to classify the known self-adaptive protocol algorithms, along with their properties. During this phase will also research, document, and develop "workload models" for highly dynamic fault-tolerant network environments. Such workload models will serve as benchmark data sets against which we will characterize the performance of existing discovery technologies and test new self-adaptive control algorithms. We will use this research and analysis to select the specific fault-tolerant protocols and mechanisms that we will focus on for the remainder of the project. We will also publish a technical report on our findings from this stage.

During the design and evaluation phase, we will devise specific self-adaptive algorithms to insert within selected dynamic discovery protocols. We will construct models of the selected protocols, and implement those models in JavaSim, a publicly available networking simulation environment developed at Ohio State University [JSIM01]. We will insert our self-adaptive algorithms into the simulation models and conduct simulation experiments using our workload models to evaluate, adjust, and tune our proposed self-adaptive algorithms. We expect several conference or journal papers to result from this phase of the project. In addition, we will work to incorporate the basic models we create for discovery protocols and for workloads into JavaSim for inclusion in subsequent public releases of the simulation environment.

During the implementation phase, we will implement the best of our self-adaptive algorithms within publicly available prototype implementations of the appropriate discovery protocols. The implication is that we will be modifying existing publicly available code, and not implementing the full base protocol ourselves. We will construct experimenter's toolkits that include software to emulate the behavior of highly dynamic, fault-tolerant, networking environments using live protocol exchanges. We will also use these implementations to validate our simulation experiments, conducted during the design and evaluation phase. We will endeavor to have our modifications included into subsequent releases of the public prototypes and will publicly release our test and measurement tools.

During the influence phase, we will draft proposals to include the most successful of our self-adaptive algorithms into the public specifications for the appropriate dynamic discovery protocols. We will present these proposals to the appropriate industry forums (e.g., IETF, Jini Community, UPnP Forum) in an effort to have them included in subsequent versions of the specifications. We also will attempt to influence the fundamental design of discovery and dynamic composition technologies so that they can become more amenable to incorporation of self-adaptive algorithms. We expect that this work will result in a paper on design principles for achieving optimal performance within fault-tolerant networks that rely upon discovery-based, component architectures.

**Milestones and Deliverables**

**September 2001**. Report analyzing discovery-based fault-tolerant networking protocols, identifying protocol mechanisms that might benefit from self-adaptive algorithms, and recommending which mechanisms should be used for the design and evaluation phase of the project.

**April 2002**. Descriptions of selected self-adaptive algorithms, along with models of the algorithms incorporated within simulations of appropriate fault-tolerant networking protocols. Benchmark workload models of highly dynamic fault-tolerant networking environments.

**September 2002**. A publishable conference or journal paper demonstrating performance improvements and associated costs for selected self-adaptive algorithms simulated within models of discovery-based fault-tolerant networking protocols.

**April 2003**. An initial prototype implementation of the best self-adaptive algorithms designed in previous stages of the project. The implementation will be incorporated within publicly available software for the appropriate fault-tolerant networking protocols. Experimenter's toolkits, including software to emulate the behavior of highly dynamic fault-tolerant networking environments using live protocol exchanges.

**September 2003**. Draft submissions to the IETF and other industry forums that propose self-adaptive algorithms for appropriate discovery technologies.

**April 2004**. Final report that describes the accomplishments of the project, and that suggests future research.

**Funding and Staff**

I omitted this part.

## Principal Investigators

Kevin Mills and Doug Montgomery
NIST
Building 820 Room 445
Gaithersburg, Maryland 20899
Tel: (301) 975-3600
Fax: (301) 590-0932
Email: kmills@nist.gov and dougm@nist.gov

## Administrative Contact

Pam Davis
NIST
Building 820 Room 612
Gaithersburg, Maryland 20899
Tel: (301) 975-2735
Fax: (301) 527-6395
Email: pamela.davis@nist.gov

## References

[JINI00] K. Arnold, et al., “JiniTM Technology Core Platform Specification”, V1.1, Sun Microsystems, October 2000.

[JSIM01] H-Y Tyan and C-J Hou, "JavaSim: A component-based compositional network simulation environment," to appear in *Western Simulation Multiconference -- Communication Networks And Distributed Systems Modeling And Simulation*, January 2001.

[OWING00] B. Bieber and J. Carpenter / ISD C4I; “Openwings A Service-Oriented Component Architecture for Self-Forming, Self-Healing, Network-Centric Systems.” http://www.openwings.org/download/specs/openwingswp.pdf

[SLP99] E. Guttman, C. Perkins, J, Veizades, and M. Day, “Service Location Protocol, Version 2.0”, RFC 2608, IETF, June 1999.

[SSDP99] Y.Y. Goland, T. Cai, P. Leach, and Y. Gu, “Simple Service Discovery Protocol/1.0: Operating without an Arbiter”, IETF Internet Draft, October 1999.

[UPNP00] Universal Plug and Play Device Architecture, Version 1.0, June 2000.